Metal or Rock?

Table of Contents

Import Libraries and Load Data

In [1]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from time import time
```

In [2]:
```
df = pd.read_csv('./Data/sonar_all_data.csv', header = None)
df = df.sample(frac = 1)
data = df.loc[:, 0:df.shape[1]-2].to_numpy()
labels = pd.get_dummies(df[df.shape[1] -1])['M'].to_numpy()
```

In [3]:
```
 data_train, data_test, labels_train, labels_test = train_test_split(data,
                                                                     labels,
                                                                     test_size=0.1,
                                                                     random_state=42)
```

Decision Tree Classifier

In [4]:
```
clf = DecisionTreeClassifier(min_samples_split=2)

t0 = time()
clf.fit(data_train, labels_train)
print("Training Time: ", round(time()-t0, 3), "s")
```

```
Training Time:  0.005 s
```

In [5]:
```
t0 = time()
preds = clf.predict(data_test)
print("Predicting Time: ", round(time()-t0, 3), "s")
print(f"Accuracy: {accuracy_score(labels_test, preds)}")
```

```
Predicting Time:  0.001 s
Accuracy: 0.7142857142857143
```

Support Vector Classifier

In [6]:
```
clf = SVC(kernel = 'rbf', C = 10000)
t0 = time()
clf.fit(data_train, labels_train)
print(f'Finished fitting data in {round(time() - t0, 3)} seconds')
```

```
Finished fitting data in 0.004 seconds
```

In [7]:
```
t0 = time()
preds = clf.predict(data_test)
print(f'Finished predicting test data in {round(time() - t0, 3)} seconds with an accuracy of {round(accuracy_sc
```

```
Finished predicting test data in 0.001 seconds with an accuracy of 0.905
```

Random Forest Classifier

In [8]:
```
clf = RandomForestClassifier(n_estimators = 200,
                             min_samples_split = 2,
                             verbose = 1)

t0 = time()
clf.fit(data_train, labels_train)
print(f'Finished fitting data in {round(time() - t0, 3)} seconds')
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
Finished fitting data in 0.363 seconds
[Parallel(n_jobs=1)]: Done 200 out of 200 | elapsed:    0.2s finished
```

In [9]:
```
t0 = time()
preds = clf.predict(data_test)
print(f'Finished predicting test data in {round(time() - t0, 3)} seconds with an accuracy of {round(accuracy_sc
```

```
Finished predicting test data in 0.028 seconds with an accuracy of 0.857
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done 200 out of 200 | elapsed:    0.0s finished
```

Out of the three classification algorithms we have considered, the Support Vector Classifier performs the best with a test data accuracy of 90.5% followed by the Random Forest Classifier at 85.7% and Decision Tree Classifier at 71.4%. In terms of training speed, clearly the Support Vector Classifier is the fastest and the prediction speed of both the Support Vector Classifier and the Decision Trees Model is approximately the same. This leads to the conclusion that for this application, the Support Vector Classifier is ideal.